Einführung in die Computerlinguistik
Statistical Natural Language Processing

Hinrich Schütze & Robert Zangenfeind

Centrum für Informations- und Sprachverarbeitung, LMU München

2015-11-09

Take-away

- Statistical Natural Language Processing (StatNLP): Introduction
- Noisy channel model: early work was based on understanding StatNLP as “decoding messages”
- Language models: probability models that distinguish more vs less likely word sequences

Outline

Unser Plan

- Teilgebiete der Linguistik
  - Phonetik und Phonologie
  - Morphologie
  - Syntax
  - Semantik
  - Pragmatik
- Statistische Sprachverarbeitung

Outline

Statistical Natural Language Processing

Definition

Statistical Natural Language Processing (StatNLP) uses methods of supervised, semisupervised and unsupervised learning to address tasks that involve written or spoken (human) language.

What does “statistical” mean?

Adjective for “statistics”

statistics = the practice or science of collecting and analyzing numerical data

Statistical parameter estimation

an important / the most important subfield of machine learning
also a subject of statistics, but the emphasis is different

Typical StatNLP applications

- automatic summarization of text
- sentiment analysis (e.g., find all *negative* reviews of the smartphone I want to buy)
- information extraction from text (e.g., find all inhibitors of a particular gene)
- machine translation

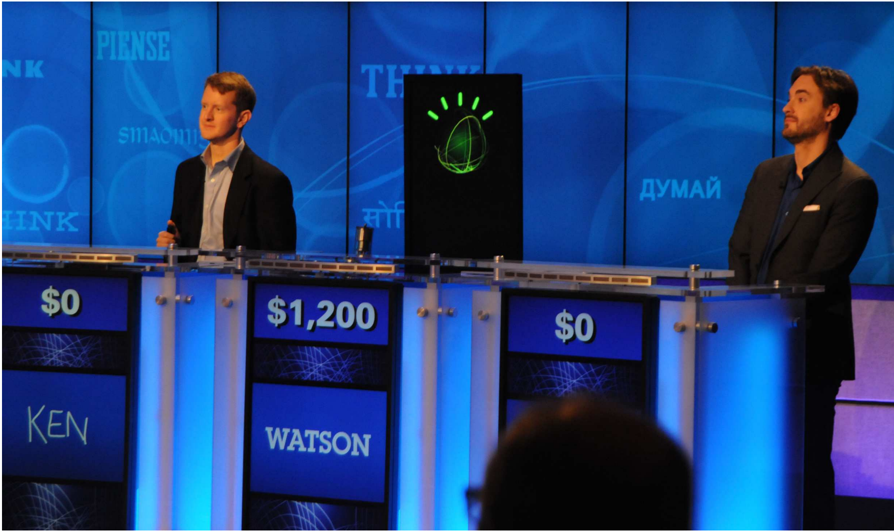
Applications that use some StatNLP

speech recognition optical character recognition information retrieval

History of StatNLP (simplified)

- 1940s, early 1950s: language as sequential process, Markov models
- 1950s, 1960s: Chomsky; statistical methods are viewed as inadequate for language.
- 1970s, 1980s: very little academic research on StatNLP, but IBM Watson group does seminal work
- 1990s: IBM Watson paradigm is adopted by computational linguists and becomes dominant approach to natural language processing.
- 2000s: The field splits methodologically into three communities.
  - traditional computational linguistics
  - a large group of researchers that use simple statistical methods
  - a small group of researchers that do active research on machine learning methods

Recent big success story 1



Recent big success story 2

Recent big success story 3

Google Translate – more on this later

Outline

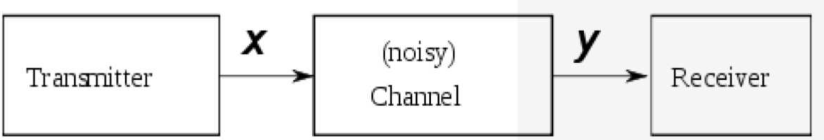
Fred Jelinek

IBM Watson approach to NLP

- sequence model
- in most cases: given an observation, select the most likely sequence that caused the observation
- We will only consider word sequences for now.

$$
\begin{aligned}
& \text{argmax}_{\text{word-sequence}} P(\text{word-sequence}|\text{evidence}) \\
= \quad & \text{argmax}_{\text{word-sequence}} \frac{P(\text{evidence}|\text{word-sequence}) P(\text{word-sequence})}{P(\text{evidence})} \\
= \quad & \text{argmax}_{\text{word-sequence}} \quad P(\text{evidence}|\text{word-sequence}) \quad P(\text{word-sequence})
\end{aligned}
$$

Noisy channel



Well-known examples of applications of noisy channel model?

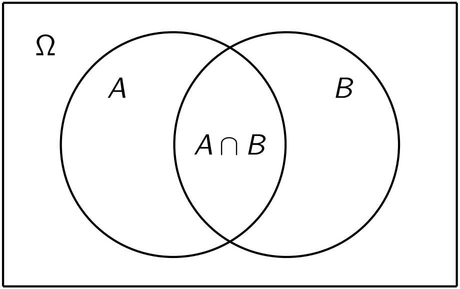
Noisy (actually, non-noisy) channel example: T9

Decode 788884278

Conditional probability

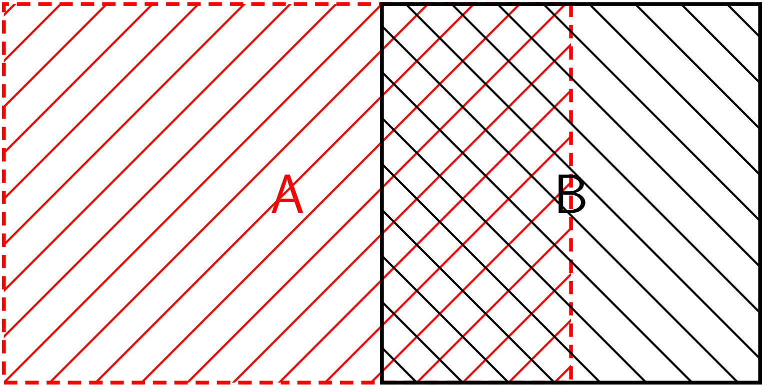
- The conditional probability is the updated probability of an event given some knowledge.
- Definition: $P(A|B) = \frac{P(A \cap B)}{P(B)}$ $(P(B) > 0)$

Venn diagram



To compute $P(A|B)$: Divide the area of $A \cap B$ by the area of B. $P(A|B) = P(A \cap B)/P(B)$
$P(B|A) = P(A \cap B)/P(A)$

Exercise



Compute $P(A|B) = P(AB)/P(B)$ and $P(B|A) = P(AB)/P(A)$

Bayes' theorem

- $P(B|A) = \frac{P(B \cap A)}{P(A)} = \frac{P(A|B)P(B)}{P(A)}$
- Or: $P(B|A) = \frac{P(A|B)P(B)}{P(A|B)P(B) + P(A|\overline{B})P(\overline{B})}$
- Follows from
  $P(A) = P(A \cap B) + P(A \cap \overline{B}) = P(A|B)P(B) + P(A|\overline{B})P(\overline{B})$

Independence

- Two events A and B are independent iff $P(A \cap B) = P(A)P(B)$
- If I learn that A is true, then that doesn't change my assessment of the probability of B (and vice versa).
- $P(A) = P(A|B)$, $P(B) = P(B|A)$

Testing for independence

- Estimate $P(A)$, $P(B)$, $P(AB)$
- Simplest way of doing this: relative frequency: $P(A) = \frac{\text{count}(A)}{\text{count}(\textit{everything})}$
- Then: Compare $P(A)P(B)$ with $P(AB)$
- Recall: A, B independent iff $P(A \cap B) = P(A)P(B)$
- If I learn that A is true, then that doesn't change
- $P(AB) \gg P(A)P(B)$: This indicates A and B are strongly dependent.
- $P(AB) \approx P(A)P(B)$: This indicates A and B are independent.
- $P(AB) \ll P(A)P(B)$: This indicates A and B are strongly dependent (negatively correlated).
- Why $\approx$?

Testing for independence: Example

A = champagne, B = sparkling

Testing for independence: Exercise

(a) compute numbers for a pair of words of your choice; (b) find two independent words

Cooccurrence counts of the words tree and missile. E.g., there are 10 documents that contain both tree and missile; there are 100 documents that contain missile and do not contain tree.

	tree	not tree
missile	10	100
not missile	1000	?

Replace the question mark in the table by a number that makes the two words independent of each other.

IBM Watson approach to NLP

- sequence model
- in most cases: given an observation, select the most likely sequence that caused the observation
- We will only consider word sequences for now.

$$
\begin{aligned}
& \operatorname{argmax}_{\text{word-sequence}} P(\text{word-sequence}|\text{evidence}) \\
= \; & \operatorname{argmax}_{\text{word-sequence}} \frac{P(\text{evidence}|\text{word-sequence}) P(\text{word-sequence})}{P(\text{evidence})} \\
= \; & \operatorname{argmax}_{\text{word-sequence}} \quad P(\text{evidence}|\text{word-sequence}) \quad P(\text{word-sequence})
\end{aligned}
$$

Classical approach to speech recognition

$$\begin{aligned} & \text{argmax}_{\text{word-sequence}} P(\text{word-sequence}|\text{evidence}) \\ = \quad & \text{argmax}_{\text{word-sequence}} \frac{P(\text{evidence}|\text{word-sequence}) P(\text{word-sequence})}{P(\text{evidence})} \\ = \quad & \text{argmax}_{\text{word-sequence}} \quad P(\text{evidence}|\text{word-sequence}) \quad P(\text{word-sequence}) \end{aligned}$$

- word sequence: sequence of words
- evidence: acoustic signal
- P(evidence|word-sequence): a model of how humans translate a sequence of (written) words into acoustics

Classical approach to optical character recognition

$$
\begin{aligned}
& \text{argmax}_{\text{word-sequence}} P(\text{word-sequence}|\text{evidence}) \\
= \quad & \text{argmax}_{\text{word-sequence}} \frac{P(\text{evidence}|\text{word-sequence}) P(\text{word-sequence})}{P(\text{evidence})} \\
= \quad & \text{argmax}_{\text{word-sequence}} \quad P(\text{evidence}|\text{word-sequence}) \quad P(\text{word-sequence})
\end{aligned}
$$

- word sequence: sequence of words
- evidence: image
- P(evidence|word-sequence): a model of how a machine (e.g., a desktop printer) translates a sequence of words into printed letters/symbols

Exercise

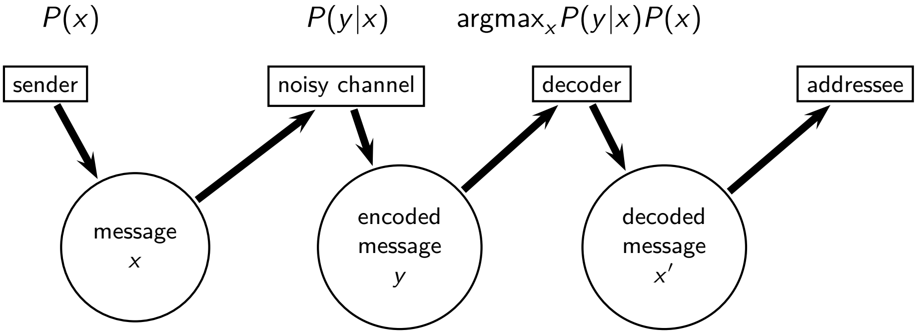
Build a noisy channel machine translation model for translating French into English.

Classical approach to machine translation (French→English)

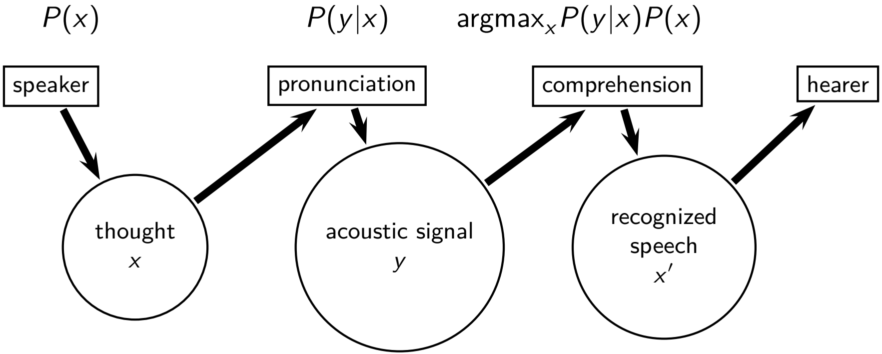
$$
\begin{aligned}
& \mathrm{argmax}_{\text{word-sequence}} P(\text{word-sequence}|\text{evidence}) \\
= \quad & \mathrm{argmax}_{\text{word-sequence}} \frac{P(\text{evidence}|\text{word-sequence}) P(\text{word-sequence})}{P(\text{evidence})} \\
= \quad & \mathrm{argmax}_{\text{word-sequence}} \quad P(\text{evidence}|\text{word-sequence}) \quad P(\text{word-sequence})
\end{aligned}
$$

- word sequence: sequence of English words
- evidence: sequence of French words
- P(evidence|word-sequence): a model of how humans translate a sequence of English words into a sequence of French words

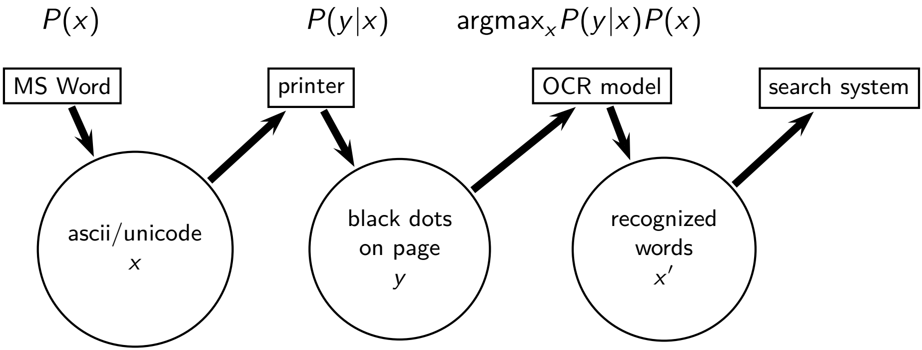
Noisy channel: Information theory / telecommunications



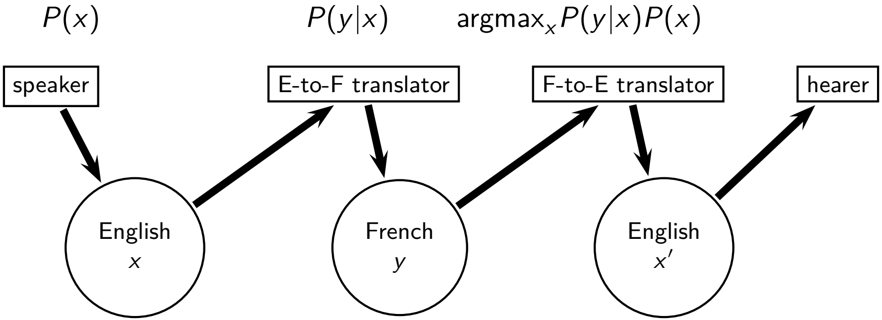
Noisy channel: Speech recognition



Noisy channel: Optical character recognition



Noisy channel: French-to-English machine translation



The two key components of the model

$$\begin{aligned}
& \operatorname{argmax}_{\text{word-sequence}} P(\text{word-sequence}|\text{evidence}) \\
= \; & \operatorname{argmax}_{\text{word-sequence}} \frac{P(\text{evidence}|\text{word-sequence}) P(\text{word-sequence})}{P(\text{evidence})} \\
= \; & \operatorname{argmax}_{\text{word-sequence}} P(\text{evidence}|\text{word-sequence}) P(\text{word-sequence})
\end{aligned}$$

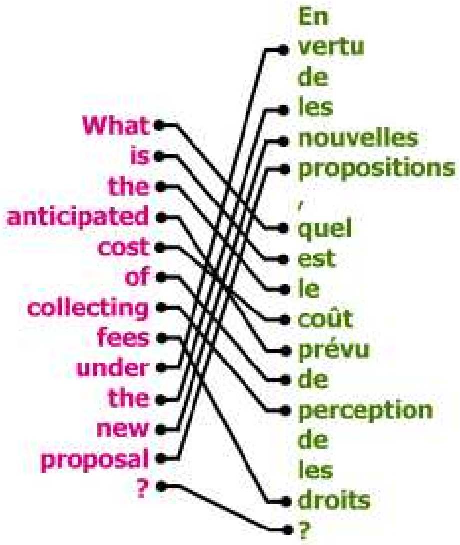
translation model

language model

How to build a translation model

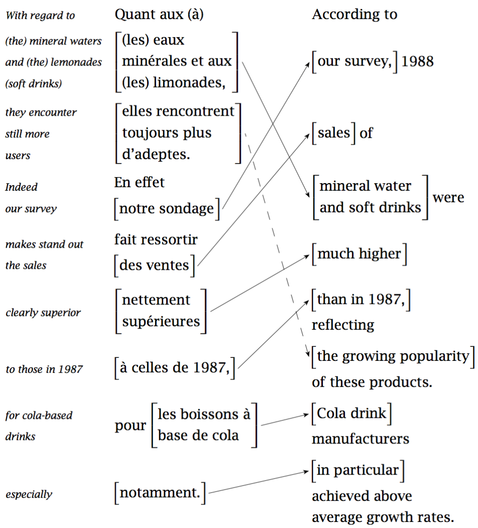
- Find a parallel corpus – a body of text where each sentence is available in two or more languages
- IBM Watson used the Canadian Hansards, the proceedings of the Canadian Parliament.
- Compute a word alignment for the parallel corpus (next slide)
- Estimate a translation model from the word alignment (that is, the model that models how humans generate French sentences from English sentences)
- Also: need a decoding/search algorithm because the search space is huge

Estimating word translation probabilities



Estimate:
$P(e_i|\text{nouvelles})$
$P(f_j|\text{fees})$
$P(f_j|\text{the})$
$P(f_j|e_0)$

"Linguistic" word/phrase alignment of a parallel corpus

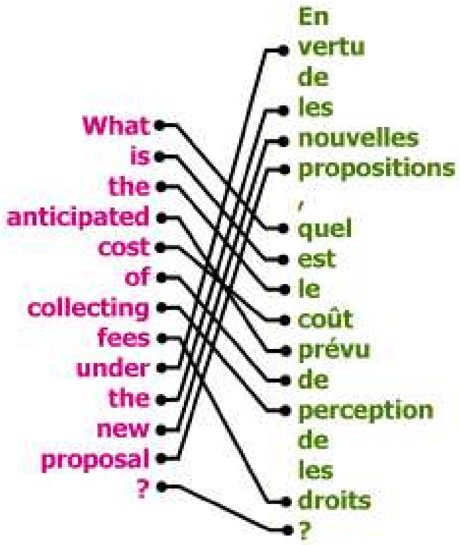


Basic translation model

$$p(f|e) \propto \sum_{a_1=0}^{l} \cdots \sum_{a_m=0}^{l} p(< a_1, \ldots, a_m >) \prod_{j=1}^{m} p(f_j|e_{a_j})$$

- e: English sentence, e_i: i^{th} word in e
- l: length of English sentence
- f: French sentence, f_j: j^{th} word in f
- m: length of French sentence
- e_{a_j} is the English word that f_j is aligned with – this assumes that the alignment is a (total) function: $a : \{1, 2, \ldots, m\} \mapsto \{0, 1, \ldots, l\}$
- There is a special word e_0, the empty cept, that all unaligned French words are aligned to.
- $p(f_j|e_{a_j})$ is the probability of e_{a_j} being translated as f_j.
- $p(< a_1, \ldots, a_m >)$ is the probability of alignment $< a_1, \ldots, a_m >$.

Estimating word translation probabilities



Estimate:
$P(e_i|\text{nouvelles})$
$P(f_j|\text{fees})$
$P(f_j|\text{the})$
$P(f_j|e_0)$

Basic translation model

$$p(f|e) \propto \sum_{a_1=0}^{l} \cdots \sum_{a_m=0}^{l} p(< a_1, \ldots, a_m >) \prod_{j=1}^{m} p(f_j|e_{a_j})$$

- e: English sentence, e_i: i^{th} word in e
- l: length of English sentence
- f: French sentence, f_j: j^{th} word in f
- m: length of French sentence
- e_{a_j} is the English word that f_j is aligned with – this assumes that the alignment is a (total) function: $a : \{1, 2, \ldots, m\} \mapsto \{0, 1, \ldots, l\}$
- There is a special word e_0, the empty cept, that all unaligned French words are aligned to.
- $p(f_j|e_{a_j})$ is the probability of e_{a_j} being translated as f_j.
- $p(< a_1, \ldots, a_m >)$ is the probability of alignment $< a_1, \ldots, a_m >$.

Formalization of alignment

e_0	e_1	e_2
empty cept	they	descended

f_1	f_2	f_3
runter	gingen	sie

a_1	a_2	a_3	a_1	a_2	a_3	a_1	a_2	a_3
0	0	0	1	0	0	2	0	0
0	0	1	1	0	1	2	0	1
0	0	2	1	0	2	2	0	2
0	1	0	1	1	0	2	1	0
0	1	1	1	1	1	2	1	1
0	1	2	1	1	2	2	1	2
0	2	0	1	2	0	2	2	0
0	2	1	1	2	1	2	2	1
0	2	2	1	2	2	2	2	2

Basic translation model

$$p(f|e) \propto \sum_{a_1=0}^{l} \cdots \sum_{a_m=0}^{l} p(< a_1, \ldots, a_m >) \prod_{j=1}^{m} p(f_j|e_{a_j})$$

- e: English sentence, e_i: i^{th} word in e
- l: length of English sentence
- f: French sentence, f_j: j^{th} word in f
- m: length of French sentence
- e_{a_j} is the English word that f_j is aligned with – this assumes that the alignment is a (total) function: $a : \{1, 2, \ldots, m\} \mapsto \{0, 1, \ldots, l\}$
- There is a special word e_0, the empty cept, that all unaligned French words are aligned to.
- $p(f_j|e_{a_j})$ is the probability of e_{a_j} being translated as f_j.
- $p(< a_1, \ldots, a_m >)$ is the probability of alignment $< a_1, \ldots, a_m >$.

Exercise

What's bad about this model? What type of linguistic phenomenon will not be translated correctly?

What's bad about this model

- Collocations, noncompositional combinations: "piece of cake"
  - Assumption violated: Each English word generates German translations independent of the other words.
- Compounds: "Kirschkuchen" vs. "cherry pie"
  - Assumption violated: For each German/French word there is a single English word reponsible for it.
- Unlikely alignments: "siehst Du" vs. "(do) you see"
  - Assumption violated: The probability of a particular alignment is independent of the words.

What's bad about this model (2)

- Morphology: "Kind" – "Kindes"
- Gender and case
- Syntax: which types of differences between German syntax and English syntax could be a problem?

Google Translate

Exercise

Devise a set of rules that will translate words like *flanierst*, *garniert*, and *Kiefer* correctly. Devise a set of rules that will handle case correctly. Devise a set of rules that will handle long-distance dependencies correctly.

Refinements

- $p(< a_0, \ldots, a_m >)$: penalize "distortions" – alignments where a word at the beginning of the sentence is translated as last word etc.
- $p(< a_0, \ldots, a_m >)$: estimate a "fertility" for each English word (the number of French words it generates on average) and penalize alignments that deviate from this fertility.
- For example, most English words generate one word, some generate two words (*farmers* $\rightarrow$ *les agriculteurs*). Penalize an alignment in which a single English word generates 10 French words.
- Phrase alignment instead of word alignment

Current research areas in statistical machine translation

- Morphology
- Syntax,
  linguistic syntax as well as data-driven automatic bracketing
- More complex nonlinear, nonsequential translation models
- Cheap acquisition of parallel corpora
  – or at least "comparable" corpora
- Scalability
- Deep learning

Outline

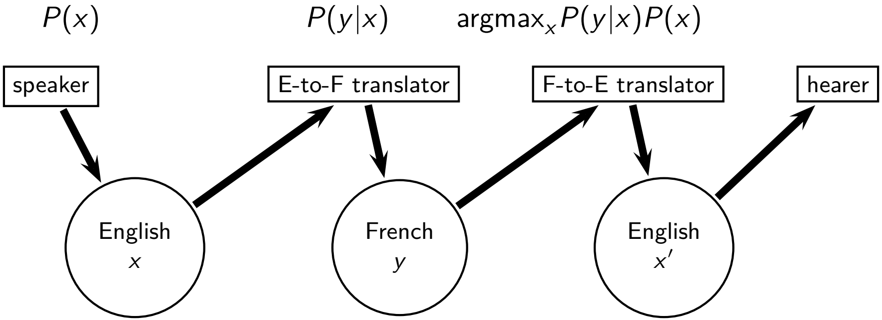
The two key components of the model

$$
\begin{aligned}
& \text{argmax}_{\text{word-sequence}} P(\text{word-sequence}|\text{evidence}) \\
= \quad & \text{argmax}_{\text{word-sequence}} \frac{P(\text{evidence}|\text{word-sequence}) P(\text{word-sequence})}{P(\text{evidence})} \\
= \quad & \text{argmax}_{\text{word-sequence}} P(\text{evidence}|\text{word-sequence}) P(\text{word-sequence})
\end{aligned}
$$

translation model

language model

Noisy channel: French-to-English machine translation



Why the language model is important

- Classical example from speech recognition
- The following two are almost indistinguishable acoustically.
- "wreck a nice beach"
- "recognize speech"
- If we had only the translation model $P(y|x)$,
  then we would not be able to make a good decision.
- We need the language model for this decision.
- $P(\text{"wreck a nice beach"}) \ll P(\text{"recognize speech"})$
- We'll choose "recognize speech" based on this.

Bigram language model

$$P(w_{1,2,\ldots,n}) = P(w_1)\prod_{i=2}^{n} P(w_i|w_{i-1})$$

- Key problem: How do we estimate the parameters?
- $P(w_1)$?
- $P(w_i|w_{i-1})$?

Maximum likelihood = Relative frequency

$$p_{ML}(w_2|w_1) = \frac{C(w_1 w_2)}{C(w_1)}$$

where $C(e)$ is the number of times the event e occurred in the training set. Example:

$$p_{\mathrm{ML}}(\text{be}|\text{would}) = \frac{C(\text{would be})}{C(\text{would})} = \frac{18454}{83735} \approx 0.22$$

Why maximum likelihood does not work

- Suppose that "Dr." and "Cooper" are frequent in our corpus. Frequency of "Dr." = 10000
- But suppose that the sequence "Dr. Cooper" does not occur in the corpus.
- What is the maximum likelihood estimate of $P(\text{Cooper}|\text{Dr.})$?
- $$p_{ML}(\text{Cooper}|\text{Dr.}) = \frac{C(\text{Dr. Cooper})}{C(\text{Dr.})} = \frac{0}{10000} = 0$$
- This means that in machine translation, any English sentence containing "Dr. Cooper" would be deemed impossible and could not be output by the translator.
- This problem is called sparseness.
- Ideally, we would need knowledge about events and their probability that never occurred in our training corpus.

Laplace

$$p_L(w_2|w_1) = \frac{C(w_1 w_2) + 1}{C(w_1) + |V|}$$

where $C(e)$ is the number of times the event e occurred in the training set, V is the vocabulary of the training set and $w_{i,j}$ is the sequence of words $w_i, w_{i+1}, \ldots, w_{j-1}, w_j$. Better estimator:

$$p_L(\text{Cooper}|\text{Dr.}) = \frac{0 + 1}{10000 + 256873} \approx 0.0000037 \rangle 0$$

So now our machine translation system has a chance of finding a good English translation that contains the phrase "Dr. Cooper".

Laplace: Better, but not great

$$p_{\mathrm{ML}}(\mathsf{be}|\mathsf{would}) = \frac{C(\mathsf{would\ be})}{C(\mathsf{would})} = \frac{18454}{83735} \approx 0.22$$

$$p_L(\mathsf{be}|\mathsf{would}) = \frac{18454 + 1}{83735 + 256873} \approx 0.05$$

Exercise

the three women saw the small mountain behind the large mountain Compute maximum likelihood and laplace estimates for:

$p(\text{three}|\text{the})$ and $p(\text{saw}|\text{the})$

Order of language models

- We have looked at a bigram model, order $= 2$.
- In many applications, notably in machine translation, language models of higher order are used: 7-gram, 8-gram, 9-gram models.

Take-away

- Statistical Natural Language Processing (StatNLP): Introduction
- Noisy channel model: early work was based on understanding StatNLP as "decoding messages"
- Language models: probability models that distinguish more vs less likely word sequences